Certification of Public Keys within an Identity Based System

L. Chen[1], K. Harrison[1], A. Moss[2], D. Soldera[1], and N.P. Smart[2]

[1] Hewlett-Packard Laboratories,
Filton Road,
Stoke Gifford,
Bristol, BS34 8QZ,
United Kingdom.
{liqun_chen, keith_harrison, david_soldera}@hpl.hp.com
[2] Computer Science Department,
Woodland Road,
University of Bristol,
Bristol, BS8 1UB,
United Kingdom.
{moss, nigel}@cs.bris.ac.uk

Abstract. We investigate a number of issues related to the use of multiple trust authorities in the type of identifier based cryptography enabled by the Weil and Tate pairings. An example of such a system is the Boneh and Franklin encryption scheme. We examine how to create an efficient hierarchy of multiple trust authorities and then go on to examine some application areas of pairing based cryptography.

1 Introduction

In 1984 Shamir [10] introduced the concept of identity based cryptography and proposed a signature scheme based on the RSA assumption. Over the years a number of researchers tried to propose secure and efficient identity based encryption algorithms, but with little success. This state of affairs changed in 2001 when two identity based encryption algorithms were proposed, one by Cocks [4] based on the quadratic residuosity assumption and another by Boneh and Franklin [1] based on the Weil pairing, although using a variant based on the Tate pairing is more efficient. A number of identity based signature algorithms based on the Tate pairing also exist, e.g. [3], [7] and [9].

The creation of these identity based schemes made it possible to approach the problem of creating a public-key infrastructure from a new angle. To this end Horwitz and Lynn [8] describe a way of coping with a hierarchy of trust authorities using only identity-based encryption. However, this advantage of only being based on identifiers comes with a disadvantage that the proposed system only works with two levels of keys and that a certain amount of collusion leads to the whole system being broken. This paper addresses the same issues as [8] and provides a more scalable solution.

The paper is structured as follows, in the next section we set up notation and recap on a number of pairing based schemes. In section 3 we describe how a hybrid of traditional PKI and IBE may be created that could overcome some of the current scalability issues with traditional PKIs. We show how the short signature of Boneh, Lynn and Shacham [2] lends itself naturally to use in the certificates of a hybrid traditional-PKI/IBE system. In section 4 we describe a scalable hierarchy of trust authorities using a combination of IBE and short signatures. In section 5 we look at some applications that would require a scalable and efficient hierarchy of trust authorities. Section 6 draws some conclusions from this paper.

2 Notation and Pairing Based Schemes

2.1 The Tate Pairing

Let G_1 and G_2 denote two groups of prime order q in which the discrete logarithm problem is believed to be hard and for which there exists a computable bilinear map

$$t : G_1 \times G_1 \longrightarrow G_2.$$

We shall write G_1 with an additive notation and G_2 with a multiplicative notation, since in real life G_1 will be the group of points on an elliptic curve and G_2 will denote a subgroup of the multiplicative group of a finite field.

Since the mapping is bilinear, we can move exponents/multipliers around at will. For example if $a, b, c \in \mathrm{F}_q$ and $P, Q \in G_1$ then we have

$$\begin{aligned} t(aP, bQ)^c &= t(aP, cQ)^b = t(bP, cQ)^a = t(bP, aQ)^c = t(cP, aQ)^b = t(cP, bQ)^a \\ &= t(abP, Q)^c = t(abP, cQ) = t(P, abQ)^c = t(cP, abQ) \\ &= \ldots \\ &= t(abcP, Q) = t(P, abcQ) = t(P, Q)^{abc} \end{aligned}$$

These tricks will be used repeatedly throughout this document.

We define the following cryptographic hash functions

$$\begin{aligned} H_1 &: \{0,1\}^* \longrightarrow G_1, \\ H_2 &: \{0,1\}^* \longrightarrow \mathrm{F}_q, \\ H_3 &: G_2 \longrightarrow \{0,1\}^*. \end{aligned}$$

2.2 Types of Public/Private Key Pairs

We require the following two types of keys:

- A standard public/private key pair is a pair (R, s) where $R \in G_1$ and $s \in \mathrm{F}_q$ with
$$R = sP$$
for some given fixed point $P \in G_1$.

- An identifier based key pair is a pair $(Q_{\mathtt{ID}}, S_{\mathtt{ID}})$ where $Q_{\mathtt{ID}}, S_{\mathtt{ID}} \in G_1$ and there is some trust authority (TA) with a standard public/private key pair given by $(R_{\mathtt{TA}}, s)$, such that the key pair of the trust authority and the key pair of the identifier are linked via

$$S_{\mathtt{ID}} = sQ_{\mathtt{ID}} \text{ and } Q_{\mathtt{ID}} = H_1(\mathtt{ID}),$$

where ID is the identifier string.

2.3 Cryptographic Primitives

We recap on the following three cryptographic primitives.

Short Signatures This scheme, due to Boneh, Lynn and Shacham [2], allows the holder of the private part s of a standard public/private key pair to sign a bit string. Let m denote the message to be signed

- **Signing :**
  Compute $V = sH_1(m)$.
- **Verification :**
  Check whether the following equation holds

$$t(P, V) = t(R, H_1(m))$$

Since this is the first time we have used the pairing it is worth demonstrating why the equation should hold for a valid signature.

$$\begin{aligned} t(P,V) &= t(P, sH_1(m)) && \text{Since } V = sH_1(m), \\ &= t(P, H_1(m))^s && \text{By linearity in the second coordinate,} \\ &= t(sP, H_1(m)) && \text{By linearity in the first coordinate,} \\ &= t(R, H_1(m)) && \text{Since } R = sP. \end{aligned}$$

Identifier Based Encryption This scheme, due to Boneh and Franklin [1], allows the holder of the private part $S_{\mathtt{ID}}$ of an identifier based key pair to decrypt a message sent to her under the public part $Q_{\mathtt{ID}}$. We present only the simple scheme which is only ID-OWE, for an ID-CCA scheme one applies the Fujisaki-Okamoto transformation [6]. Let m denote the message to be encrypted then:

- **Encryption :**
  Compute $U = rP$ where r is a random element of F_q. Then compute

$$V = m \oplus H_3(t(R_{\mathtt{TA}}, rQ_{\mathtt{ID}}))$$

  Output the ciphertext (U, V).

- **Decryption :**
  Decryption is performed by computing

$$\begin{aligned} V \oplus H_3(t(U, S_{\mathrm{ID}})) &= V \oplus H_3(t(rP, sQ_{\mathrm{ID}})) \\ &= V \oplus H_3(t(P, Q_{\mathrm{ID}})^{rs}) \\ &= V \oplus H_3(t(sP, rQ_{\mathrm{ID}})) \\ &= V \oplus H_3(t(R_{\mathrm{TA}}, rQ_{\mathrm{ID}})) \\ &= m. \end{aligned}$$

Identifier Based Signatures There are a number of such schemes based on the Tate pairing, we present the one due to Hess [7], which is not only efficient but has a security proof relative to the computational Diffie-Hellman problem in G_1. Let m denote the message to be signed then:

- **Signing :**
  Compute $r = t(P, P)^k$ where k is a random element of F_q. Apply the hash function H_2 to $m\|r$ to obtain $h = H_2(m\|r)$. Compute

$$U = hS_{\mathrm{ID}} + kP.$$

  Output (U, h) as the signature on the message m.
- **Verification :**
  Compute

$$r = t(U, P) \cdot t(Q_{\mathrm{ID}}, -R_{\mathrm{TA}})^h.$$

  Accept the signature if and only if $h = H_2(m\|r)$.

3 A Hybrid PKI/IBE

3.1 Combining a traditional PKI with IBE

In [8] a way of coping with a hierarchy of trust authorities is presented. The approach is for trust authorities to produce keys for trust authorities further down the hierarchy. There is only one standard public/private key pair in the whole system, which belongs to the root trust authority. All other keys are identifier based key pairs and as such the hierarchy produced can be considered as a pure identifier based infrastructure.

However, this advantage of only being based on identifiers comes with a disadvantage that the proposed system only works with two levels of keys and that a certain amount of collusion leads to the whole system being broken. Thus this type of hierarchical approach as a means to replacing traditional, X509 like, PKI systems seems to suffer from worse scalability issues than X509.

As a solution we propose a hybrid system which merges traditional PKI solutions with identifier based solutions. We assume a multitude of standard public/private key pairs held by trust authorities, with user keys being identity based. A number of points need to be made

- This model of multiple trust authorities is more likely to resemble the "real world" where no global hierarchy is in place.
- We do not assume that trust authorities are embodied in reputable organisations, for example some applications may want trust authorities to be embedded into ones PDA. However, we do not exclude global trust authorities such as Verisign or Microsoft in our model.

Hence, multiple trust authorities can exist but we assume encryption and signatures are made using identifier based keys. With multiple trust authorities one of course needs some way of authenticating, or cross certifying, the authorities as in a traditional form of PKI solution.

So has this hybrid PKI/IBE based solution bought us anything? It appears to have created a level of complication, but our belief is that this makes the system more scalable. A common problem with traditional PKI is that whilst it is very good at authenticating domain names, as in the use of SSL, it is rather poor in authenticating large numbers of individual users. On the other hand, identifier based systems are very good at identifying individual users, but poor when it comes to multiple trust domains (as the paper [8] demonstrates).

We use two analogies for the state of affairs we envisage:

- The first is from the world of telecommunications. In this world there are two systems (often run by two separate companies). There is the *local loop* which is the copper wires (or fibre optic cables) from your home to the exchange and then there is the global long distance telephone system. One should think of the local loop being identifier based and the long distance network being PKI based.
- The email system has a similar discontinuity between what are essentially local and global names. Take the email address

  `Alice@people.iacr.org`

  The `people.iacr.org` part is a global name which can be authenticated efficiently using standard certificate chains. The problems arise when one tries to push down the PKI solution to the next level. The `Alice` part is therefore more easily dealt with using identifier based systems.

3.2 Certificates using Short Signatures

We examine this hierarchy of TA's in more detail using the above email address as an example. We let the three TA's each with their own standard public/private key pairs.

Entity	Private Key	Public Key
org	s_1	$R_{\texttt{org}} = s_1 P$
iacr	s_2	$R_{\texttt{iacr}} = s_2 P$
people	s_3	$R_{\texttt{people}} = s_3 P$

The entity Alice is issued an identifier based key pair from the trust authority people, namely

$$S_{\texttt{Alice}} = s_3 Q_{\texttt{Alice}} \text{ and } Q_{\texttt{Alice}} = H_1(\texttt{Alice}).$$

Now suppose someone, Bob, wishes to send the entity denoted by

Alice@people.iacr.org

an email or someone wishes to verify that entity's signature. Bob first needs to obtain a trusted copy of the public key of the people trust authority.

Supposing Bob already trusts the public key of the trust authority org. They could simply verify a certificate chain down to the public key of people using standard certificate formats, but since each trust authority has the correct type of standard public/private key pair we can use the short signature scheme of Boneh, Lynn and Shacham.

The certificate of the trust authority iacr, as produced by the trust authority org would then look like

$$(\texttt{Subject}, \texttt{Issuer}, \texttt{Key}, \texttt{Signature}) = (\texttt{iacr}, \texttt{org}, R_{\texttt{iacr}}, V)$$

where

$$V = s_1 H_1(R_{\texttt{iacr}} \| \texttt{iacr}).$$

Note that we can use the same code to perform certificate checking and verification as one would use to produce identifier based encryption and signatures. This will be an advantage on small devices which may only allow small code footprints.

4 Hierarchies of Linked Trust Authorities

In the previous section we assumed that the various trust authorities were not linked via their identities but simply had traditional public/private key pairs. In this section we examine what happens when the trust authorities are linked in an identifier based hierarchy.

Entity	Standard Private Key	Standard Public Key	ID Based Private Key	ID Based Public Key
org	s_1	$R_{\texttt{org}} = s_1 P$	-	-
iacr	s_2	$R_{\texttt{iacr}} = s_2 P$	$S_{\texttt{iacr}} = s_1 Q_{\texttt{iacr}}$	$Q_{\texttt{iacr}} = H_1(\texttt{iacr})$
people	s_3	$R_{\texttt{people}} = s_3 P$	$S_{\texttt{people}} = s_2 Q_{\texttt{people}}$	$Q_{\texttt{people}} = H_1(\texttt{people})$
Alice	-	-	$S_{\texttt{Alice}} = s_3 Q_{\texttt{Alice}}$	$Q_{\texttt{Alice}} = H_1(\texttt{Alice})$

We now have the chance to authenticate public keys in another way. If a user trusts the standard public key of any TA in the hierarchy (either org, iacr or people), this TA will become the root of trust to the user; and then this TA's

corresponding standard private key will become the master key to the user. In this section, we consider an example where a user trusts the standard public key of iacr, i.e. s_2P. Based on this trust he wishes to authenticate the identifier based public key of Alice, which has been issued by the trust authority people. We shall see that both iacr and people are able to offer this authentication service.

4.1 Transferring Trust at the upper level

One obvious way of doing this is for iacr to sign people's standard public key, using the short signature scheme, as above. Since iacr's standard public key is trusted, the verifier will then trust people's standard public key and so will trust Alice's identifier based public key.

We can represent this transfer of trust from iacr to people via

$$\texttt{iacr} \longrightarrow \texttt{people},$$

since the certification is performed by iacr.

4.2 Transferring Trust at the lower level

There is another obvious solution, which is for people to sign its own standard public key using an identifier based signature scheme, with the identifier based private key supplied to people by iacr. Since iacr's standard public key is trusted, the verifier trusts people's identifier based public key and will then trust the signature on people's standard public key. Just as before the verifier will then trust Alice's identifier based public key.

We represent this transfer of trust from iacr to people via

$$\texttt{iacr} \longleftarrow \texttt{people},$$

since now the certification is performed by the trust authority people.

4.3 Balanced Trust Transferal

However, there is another more natural way which can be deployed by either iacr or people and which to the user is transparent as to who actually produced the authentication of people's public key. A situation which we can represent diagrammatically via

$$\texttt{iacr} \longleftrightarrow \texttt{people}.$$

The system we provide below provides implicit authentication of the key $R_{\texttt{people}}$, in that explicit authentication is only obtained once the key $R_{\texttt{people}}$ has been seen 'in action', in other words it is used to verify a signature.

The verifier is assumed to know and trust the standard public key $R_{\texttt{iacr}}$. They wish to obtain implicit trust that people's standard public key $R_{\texttt{people}}$ is linked to the entity which iacr is claiming to be people. Once this linkage is established the verifier can then use Alice's public key.

One of the following stages is then executed:

– If the authenticating party is people, then people generates a random value of $r \in \mathrm{F}_q$ and publishes

$$C_1 = rS_{\texttt{people}} = rs_2 Q_{\texttt{people}},$$
$$C_2 = rQ_{\texttt{people}},$$
$$C_3 = rR_{\texttt{people}}.$$

– If the authenticating party is iacr, then iacr generates a random value of $r \in \mathrm{F}_q$ and publishes

$$C_1 = rs_2 Q_{\texttt{people}},$$
$$C_2 = rQ_{\texttt{people}},$$
$$C_3 = rR_{\texttt{people}}.$$

The verifier can then check that the linkage is as claimed by checking the following two equations hold

$$\begin{aligned} t(C_2, R_{\texttt{people}}) &= t(rQ_{\texttt{people}}, R_{\texttt{people}}) \\ &= t(Q_{\texttt{people}}, rR_{\texttt{people}}) \\ &= t(Q_{\texttt{people}}, C_3), \\ t(P, C_1) &= t(P, rs_2 Q_{\texttt{people}}) \\ &= t(s_2 P, rQ_{\texttt{people}}) \\ &= t(R_{\texttt{iacr}}, C_2). \end{aligned}$$

These two equations demonstrate that the discrete logarithms are related as they should be.

5 Applications of Pairing Based Systems

We now examine some novel applications of the pairing based cryptosystems given at the beginning of this paper. All of the following applications assume that the trust authorities are in fact trusted by all users, and hence all require some form of certification like those proposed in the prior sections.

5.1 Delegation of Rights

Up to now when we have used identifiers they have really been identities. Traditional PKI sometimes makes a distinction between an identity certificate (represented by a 4-Tuple in SPKI for example [5]) and an authorisation certificate (represented by a 5-Tuple in SPKI). Since strings correspond to keys in an identifier based system, we can replace identity strings in our discussion above with authorisation strings.

For example given a SPKI s-expression which describes some access rights, we can write down immediately the corresponding identifier public key corresponding to this s-expression. There is no need to bind the s-expression to the key, since the s-expression *is* the key.

We now describe how delegation of authorisations can be handled. Suppose we have some trust authority (say Alice) who has control of some resource. Suppose Alice has a standard public/private key pair given by

$$R_{\texttt{Alice}} = sP.$$

Assume that Bob has a public key given by $M_{\texttt{Bob}}$, this could either be a standard public key or an identifier based one.

Alice wishes to pass authorisation to use this resource to Bob. In SPKI this is represented by the 5-tuple

`(Issuer, Subject, Delegate, Authorization, Validity),`

where the standard SPKI format is to have `Issuer` and `Subject` being hashes of public keys, `Delegate` being a `Yes/No` flag, `Authorization` being the description of what is being authorised and `Validity` being the validity period.

Alice forms the s-expression given by

$$\sigma = M_{\texttt{Bob}} \| \texttt{Delegate} \| \texttt{Authorization} \| \texttt{Validity}$$

and then forms the public/private key pair given by

$$S_\sigma = sQ_\sigma \text{ where } Q_\sigma = H_1(\sigma).$$

Alice then gives the private key S_σ to Bob.

For Bob to now use this resource he needs to demonstrate

- He knows the private key corresponding to $M_{\texttt{Bob}}$, i.e. he is Bob.
- He knows the private key corresponding to Q_σ, i.e. Alice has given him the authorisation.

To demonstrate the two facts he can either produce a signature or engage in a challenge-response protocol using the two corresponding private keys.

Now consider the case where the `Delegate` field is set to `Yes`. In this case Bob is allowed to delegate some, or all, of his authorisation from Alice to a third party, say Charlie. In this case Bob himself needs to act as a trust authority and so must have a standard public/private key pair

$$R_{\texttt{Bob}} = tP.$$

Bob can then create a delegation

$$\tau = M_{\texttt{Charlie}} \| \texttt{Delegate'} \| \texttt{Authorization'} \| \texttt{Validity'}$$

in the same way as Alice did. To use the resource Charlie needs to present both σ and τ, and needs to prove

- He knows the private key corresponding to $M_{\texttt{Charlie}}$, i.e. he is Charlie.
- He knows the private key corresponding to Q_τ, i.e. Bob has given him the authorisation.

But how is Alice's original authorisation σ going to be authenticated? First Alice should use Bob's key $R_{\texttt{Bob}}$ within the s-expression for σ, where we interpret this as saying only the key $R_{\texttt{Bob}}$ is allowed to delegate. Finally Bob needs to pass onto Charlie some information which proves that Alice gave him authorisation by revealing the value of S_σ to him. This last task is accomplished in one of two ways:

- As in the last section by Bob publishing

$$C_1 = rS_\sigma, C_2 = rQ_\sigma, C_3 = rR_{\texttt{Bob}}.$$

  The advantage of this method is that Alice could also produce this information for Charlie, however the disadvantage is the relatively large bandwidth considerations.
- Bob could sign, using the private key S_σ and the earlier identifier based signature scheme, the message given by Q_τ.

Composition of the delegated authorisations can be accomplished using the standard SPKI 5-Tuple reduction rules. Notice that, one can even remove the delegation field from the s-expressions, since to delegate we require a standard public/private key. We can bind the authorisation to a standard public key when delegations are allowed and an identifier based public key when delegations are not allowed.

5.2 Creating Groups

We give an example where the use of trust authorities, even at the user level, gives a number of added advantages when combined with the identity based encryption and signature schemes. Note, this application requires the trust authorities standard public key to itself be trusted. Hence, the following application assumes the existence of some form of certification for the trust authorities public keys as we have discussed in the rest of this paper.

Imagine a city in which there is a city wide local public wireless network. For example Bristol University, Hewlett-Packard and a number of other organisations plan to roll out such an infrastructure across Bristol in the near future. Suppose you arrive in this city for a conference and you are good friends with Alice, who lives in this city. You would ideally like Alice to pick you up from the airport, but if she is not available then one of Alice's friends you would trust to do this. Therefore you broadcast a message over the network when you land saying, "I've arrived at the airport! I am completely lost! Could one of Alice's friends pick me up?".

Clearly if this was broadcast in the clear then you would leave yourself open for any unscrupulous person to come and try and mug you. Whilst this may not

be a major problem in a relatively peaceful city, but it may be a problem in some cities. The question arises, how can you encrypt to Alice's friends when you may not know who they are?

To overcome this problem consider the situation in which people are their own trust authorities and issue keys to certain subsets of their acquaintances. In our example Alice is a trust authority and has a public/private key given by

$$R_{\texttt{Alice}} = sP.$$

She then, when she meets her friends, gives them a public/private key not according to the actual name but simply under the identifier `Friend`. Such a device to create, distribute and accept keys can be embedded into either a PDA or into some wearable computer that people in the city use to interact with the wireless network.

In this way our hapless traveller, just arrived for the conference, can encrypt a message to all of Alice's friends by using the pair of keys

$$R_{\texttt{Alice}} \text{ and } Q_{\texttt{Friends}}.$$

This encrypted message can then be broadcast to the whole city, knowing that only Alice's friends can decrypt it.

5.3 Addition of Multiple Short Signatures

Although not an application of identifier based cryptography as such, the following illustrates another advantage of the short signature scheme based on pairings. Suppose we have three users Alice, Bob and Charlie with standard public/private key pairs given by

$$R_A = aP,\ R_B = bP,\ R_C = cP.$$

Now suppose they wish to all commit to some document by signing it. For example the document could be a treaty and the three parties could be heads of state, or the document could be a will and the three parties could be the person and two witnesses.

Suppose the document is represented by the string s. They can then generate individual signatures by computing

$$\begin{aligned} V_A &= aH_1(s), \\ V_B &= bH_1(s), \\ V_C &= cH_1(s). \end{aligned}$$

However, one only needs to store a single signature given by

$$V = V_A + V_B + V_C.$$

Since this can be verified by using the "virtual" public key obtained by computing

$$R = R_A + R_B + R_C.$$

Note that in order to guarantee that the signature is unforgeable, one can verify that each entity has knowledge of their private keys.

6 Conclusion

We have shown how one can create simple certificate chains for identifier based cryptosystems using either the short signature system of Boneh, Lynn and Shacham or using the identifier based signature scheme of Hess and others. We have argued that this is more efficient than using a traditional X.509 based solution due not only to bandwidth but also because the code required to produce the certificate chains can reuse a lot of the routines needed for the end applications of identity based signatures and encryption. Thus code foot print will be smaller.

We have also given a method of certification in a hierarchy of trust authorities which can be performed either by entities certifying down the chain of trust or by entities certifying up the chain of trust. The advantage of this scheme is that it is transparent to the verifying party as to who actually performed the certification.

Finally we have examined a number of application domains of pairing based cryptography, all of which produce advantages over standard public key cryptographic systems.

References

1. D. Boneh and M. Franklin. Identity based encryption from the Weil pairing. *Advanced in Cryptology - CRYPTO 2001*, Springer-Verlag LNCS 2139, 213–229, 2001.
2. D. Boneh, B. Lynn and H. Shacham. Short signatures from the Weil pairing. *Advances in Cryptology - ASIACRYPT 2001*, Springer-Verlag LNCS 2248, 514–532, 2001.
3. J. C. Cha and J. H. Cheon. An identity-based signature from gap Diffie-Hellman groups. Preprint 2002.
4. C. Cocks. An identity based encryption scheme based on quadratic residues. *Cryptography and Coding*, Springer-Verlag LNCS 2260, 360–363, 2001.
5. C. Ellison and B. Frantz. SPKI certificate theory. Internet RFC 2693, 1999.
6. E. Fujisaki and T. Okamoto. Secure integration of asymmetric and symmetric encryption schemes. *Advances in Cryptology - CRYPTO '99*, Springer-Verlag LNCS 1666, 537–554, 1999.
7. F. Hess. Efficient identity based signature schemes based on pairings. To appear *Selected Areas in Cryptography – 2002.*
8. J. Horwitz and B. Lynn. Hierarchical identity-based encryption. *Advances in Cryptology - EUROCRYPT 2002*, Springer-Verlag LNCS 2332, 466–481, 2002.
9. K. Paterson. ID-based signatures from pairings on elliptic curves. Preprint 2002.
10. A. Shamir. Identity based cryptosystems and signature schemes. *Advanced in Cryptology - CRYPTO '84*, Springer-Verlag LNCS 196, 47–53, 1985.